VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA

EKONOMICKÁ FAKULTA

KATEDRA APLIKOVANÉ INFORMATIKY

Návrh a realizace vybrané části informačního systému

Design and Realization of the Selected Part of the Information System

Student: Václav Koběrský

Vedoucí bakalářské práce: Ing. Vítězslav Novák, Ph.D.

Ostrava 2013

VŠB - Technická univerzita Ostrava
Ekonomická fakulta
Katedra aplikované informatiky

Zadání bakalářské práce

Student: **Václav Koběrský**

Studijní program: B6209 Systémové inženýrství a informatika

Studijní obor: 6209R001 Aplikovaná informatika

Téma: Návrh a realizace vybrané části informačního systému
Design and Realization of the Selected Part of the Information System

Zásady pro vypracování:

1. Úvod
2. Teoretická východiska práce
3. Návrh informačního systému a jeho realizace
4. Zhodnocení navrhovaného řešení
5. Závěr

Seznam použité literatury
Seznam zkratek
Prohlášení o využití výsledků bakalářské práce
Seznam příloh
Přílohy

Seznam doporučené odborné literatury:

BRUCKNER, Tomáš et al. *Tvorba informačních systémů: principy, metodiky, architektury*. Praha: Grada, 2012. ISBN 978-80-247-4153-6.
CONOLLY, T., C. BEGG a R. HOLOWCZAK. *Mistrovství - databáze: profesionální průvodce tvorbou efektivních databází*. Brno: Computer Press, 2009. ISBN 978-80-251-2328-7.
KISZKA, Bogdan. *1001 tipů a triků pro jazyk Java*. Brno: Computer Press, 2009. ISBN 978-80-251-2467-3.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Vítězslav Novák, Ph.D.**

Datum zadání: 23.11.2012
Datum odevzdání: 10.05.2013

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA Ekonomická fakulta 1 OSTRAVA

Ing. Petr Rozehnal, Ph.D.
vedoucí katedry

prof. Dr. Ing. Dana Dluhošová
děkanka fakulty

Prohlašuji, že jsem celou práci, včetně všech příloh, vypracoval samostatně.

Přílohu č. 1, danou mi k dispozici, jsem samostatně doplnil.

.....

Václav Koběrský

V Ostravě dne 10.5.2013

Tímto bych chtěl poděkovat Ing. Vítězslavu Novákovi Ph.D. za odbornou pomoc a rady při vedení mé bakalářské práce.

Děkuji také mé rodině a přítelkyni za psychickou i materiální podporu.

V Ostravě 10.5.2013 Václav Koběrský

Obsah

1 Úvod

„Informační systémy a informační a komunikační technologie (IS/ICT) jsou neoddělitelnou součástí současného světa a silně ovlivňují způsob práce lidí v mnoha oblastech každodenního života.“ (Bruckner, 2012, s. 12)

Žijeme v době, kdy tyto technologie dávno nejsou doménou několika málo odborníků. Naopak, přístup k nim a hlavně prospěch z nich má dnes už téměř každý. Prostřednictvím několika málo kliknutí můžeme pohodlně získat informace, které bychom si dříve museli obstarávat jinými, složitějšími způsoby. A právě zjednodušením přístupu k informacím se bude tato práce zabývat.

Jelikož je oblast informačních systémů velmi liberální, můžeme k tvorbě či dodávce informačního systému přistoupit mnoha různými způsoby. Zároveň bychom neměli zapomenout, že jde o velmi složitou problematiku. Přestože existuje řada metodik, doporučení i ověřených praktik, kterými se lze řídit, mnoho projektů tvorby IS končí neúspěchem, protože pro člověka je poměrně snadné přistoupit k tvorbě informačního systému arogantně, s důvěrou ve vlastní a nutně zjednodušený pohled. Právě zanedbání podstatných aspektů však často vede k neúspěšným projektům a tím ke zbytečně vynaloženým penězům, času a úsilí. (Bruckner, 2012)

Proto se v této práci autor pokusí přistoupit k řešenému problému s určitým nadhledem, aby se výsledek práce co nejvíce přiblížil představám zadavatele a splnil všechny vymezené cíle.

Hlavním cílem této práce tedy bude navrhnout a vytvořit informační systém určený pro menší základní školu, který bude fungovat jako samostatná webová aplikace nebo bude zakomponován do webových stránek školy. Tento systém přehledně zobrazí přesně ty informace a data, která budou požadována. Bude sloužit zejména rodičům, kteří mají zájem prostřednictvím internetu průběžně sledovat studijní výsledky svých dětí, a to z pohodlí domova či zaměstnání, nebo kteří z důvodu časové tísně nemají možnost přijít se informovat osobně.

2 Teoretická východiska práce

2.1 Historie informačních systémů

O definici toho, co je to vlastně systém, se pokusil již Aristoteles:

„Mnohé složité věci jsou jako celek více než jen souhrn částí, ze kterých se skládají.“ (Bruckner, 2012, s. 12)

Při pohledu do historie se k systémům přistupovalo hierarchicky. Rozdělovaly se na třídy a podtřídy. Tento přístup však podle dnešní teorie nebyl dokonalý. Moderní pohled na systém se začal objevovat až v padesátých letech 20. století.

Průkopníkem tohoto přístupu byl Ludwig von Bertalanfy. Pojem systém sice přesně nedefinoval, nicméně prosazoval, že *„základem vědeckých přístupů je, že dynamický celek je nadřazen jednotlivostem jeho částí“*. Systém si představoval jako *„celek, který se skládá z částí a z interakcí mezi nimi“*. (Bruckner, 2012, s. 12).

Klasifikační hierarchický přístup byl tedy nahrazen systémovým, tzv. Bertalanfyho obecnou teorií systémů, která říká, že systém lze zkoumat naprosto stejným způsobem bez ohledu na to, o jaký systém se jedná. Zda o fyzikální, biologický či jiný.

Bertalanfy také popsal otevřený a uzavřený systém a rozdíl mezi nimi. Uzavřený systém nemá žádné vstupy ani výstupy. Otevřený systém naopak má výstupy do svého okolí a vstupy do něj samotného. (Bruckner, 2012)

Z toho vyplývá, že u otevřených systémů je důležité zkoumat jejich okolí a vliv tohoto okolí na systém samotný.

Podle mezinárodních norem ISO systémem rozumíme: *„Soubor komponent účelově uspořádaných k dosažení určitého cíle nebo skupiny cílů.“* To mohou být systémy obecné: *„Systémy vytvořené a používané lidmi, které poskytují produkt nebo službu v definovaném prostředí pro uspokojení potřeb uživatelů a ostatních zainteresovaných stran. Zahrnují hardware, software, data, lidi, procesy a procedury, zařízení, materiál a přírodní zdroje.“* Nebo softwarově intenzivní: *„Systémy, kde software hraje dominantní nebo převažující roli.“* (Bruckner, 2012, s. 13)

Následující obrázek (Obrázek_1) ukazuje pohled na systém podle Brucknera.

Obrázek_1. Systém jako celek částí s jejich vztahů (Bruckner, 2012, s. 14)

Pro dobrou orientaci a správné pochopení práce budou dále v jednotlivých kapitolách vymezeny základní pojmy. Nejprve bude popsán informační systém jako celek. Poté se plynule přejde k jeho jednotlivým částem, tzn. k databázi a použitým programovacím jazykům, ve kterých bude napsána aplikace.

2.2 Informační systém

Nejprve je potřeba uvést, co jsou to data a informace, aby vyplynul zřejmý rozdíl mezi nimi.

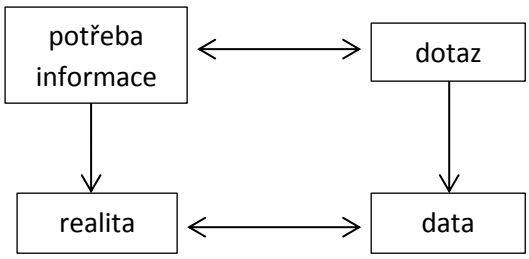
Data – *„Surová (nezpracovaná) fakta, která mají určitou důležitost pro jednotlivce nebo organizaci.“* (Conolly, 2009, s. 36)

Informace – *„Data, která prošla zpracováním nebo dostala strukturu, která jim dává pro jednotlivce nebo organizaci význam.“* (Conolly, 2009, s. 36)

Za zmínku jistě také stojí vydařená definice informačního systému podle Voříška (1997, s. 15) - *„Informační systém je soubor lidí, technických prostředků a metod zajišťujících sběr, přenos, uchování, zpracování a prezentaci dat, jehož cílem je tvorba a poskytování informací podle potřeb jejich příjemců, činných v systémech řízení.“*

„Informační systém slouží nejen ke sběru, správě a kontrole dat používaných a vytvářených organizací, ale umožňuje také transformaci dat na informace. Také poskytuje infrastrukturu pro usnadnění šíření informací k těm, kdo je potřebují.“ (Conolly, 2009, s. 108)

Pro zjednodušení a upřesnění pojmu informační systém (tak aby popis nejlépe odpovídal způsobu použití v této práci), bude nejlepší si informační systém představit jako program či aplikaci, která zpracovává data a prezentuje je již strukturované přesně podle požadavku uživatele. Přehledně je informační systém zobrazen na následujícím schématu (Obrázek_2). Další obrázek ukazuje IS podrobněji i s jeho prvky (Obrázek_3).

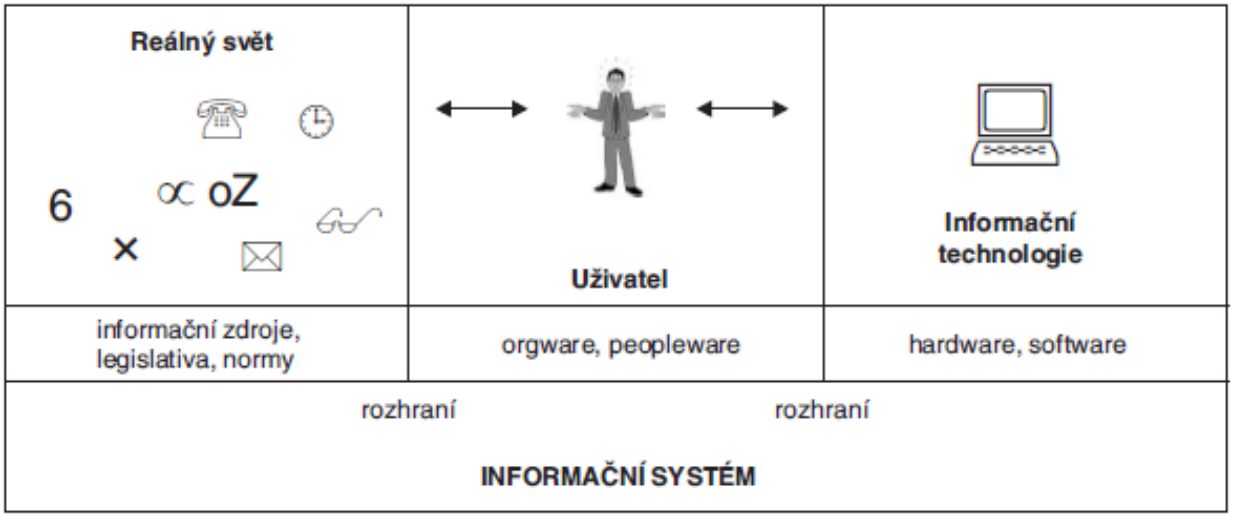


Obrázek_2. Schéma informačního systému (Zdroj: vlastní)

Komponenty IS

„Informační systém se skládá z následujících komponent:

- *technické prostředky (hardware) – počítačové systémy různého druhu a velikosti, doplněné o potřebné periferní jednotky, které jsou v případě potřeby propojeny prostřednictvím počítačové sítě a napájeny na diskový subsystém pro práci s velkými objemy dat.*
- *programové prostředky (software) – tvořené systémovými programy řídícími chod počítače, efektivní práci s daty a komunikaci počítačového systému s reálným světem, a programy aplikačními, řešícími určité třídy úloh určitých tříd uživatelů.*
- *organizační prostředky (orgware) – tvořené souborem nařízení a pravidel definujících provozování a využívání informačního systému a informačních technologií.*
- *lidská složka (peopleware) – řešení otázky adaptace a účinného fungování člověka v počítačovém prostředí, do kterého je vřazen.*
- *reálný svět (informační zdroje, legislativa, normy) – kontext informačního systému.“* (Tvrdíková, 2008, s. 19)



Obrázek_3. Prvky informačního systému. (Tvrdíková, 2008, s. 20)

2.3 Databáze

V současné době jsou databáze základní součástí informačních systémů. Jejich použití v této práci je nezbytné a proto je důležité uvést potřebnou teorii také k nim.

Co je to vlastně databáze vystihuje tato definice Andyho Oppela (2008, s. 9): *„Databáze je kolekce vzájemně souvisejících datových položek, které jsou spravovány jako jediná jednotka.“* Nebo jak uvedl Thomas Conolly (2009, s. 57): *„Databáze je sdílená kolekce logicky souvisejících dat (a popisu těchto dat), navržená pro plnění informačních potřeb organizace.“*

Pro reprezentaci objektů a událostí reálného světa se používá model dat. Jeho účelem je učinit data srozumitelnými. **Model dat** je *„Integrovaná kolekce konceptů pro popis dat, relací mezi daty a omezení používaných dat.“* (Conolly, 2009, s. 62)

„U datového modelu lze uvažovat tři složky:

1. *Strukturální část, která obsahuje množinu pravidel, jak je třeba konstruovat databázi.*
2. *Manipulační část, definující typy operací (transakcí), které jsou na datech přípustné (to zahrnuje operace pro aktualizaci nebo vyvolání dat a operace pro změnu struktury databáze).*
3. *Případně množinu pravidel integrity, která zajišťují, že data jsou přesná.“* (Conolly, 2009, s. 63)

Nejpoužívanějším je relační model dat, v němž jsou data logicky organizována do tabulek. Tento model využívá mnoho databázových produktů jako například Microsoft SQL Server, Oracle Database Management System, DB2 a Microsoft Office Access.

Pomocí modelu dat je možno data organizovat v systému řízení databáze a to pěti různými způsoby – lineárním, hierarchickým, síťovým, relačním a objektovým. Dále je vysvětlen pojem systém řízení databáze a podrobněji rozebrány jednotlivé druhy modelu dat využívané v systémech řízení.

2.3.1 Systém řízení databáze (DBMS)

„Umožňuje uživateli definovat, vytvářet a udržovat databázi a poskytuje řízený přístup k této databázi. Interaguje s uživateli, databázovými aplikacemi a s databází.“ (Conolly, 2009, s. 38, 39)

Dále tento systém umožňuje uživateli měnit a přidávat data a také se pomocí dotazovacího jazyka (například SQL) na tato data dotazovat. Dále řídí souběžný přístup k datům více uživateli (zamezuje kolizím), zajišťuje bezpečnostní mechanismy a opatření pro zálohu a obnovení databáze.

- Hierarchický DBMS využívá k uchování záznamů invertovanou stromovou strukturu. To umožňuje použití sériových paměťových zařízení, jako jsou magnetické pásky. Hierarchický systém řízení databáze se řadí mezi první generaci DBMS stejně jako síťový.
- Síťový DBMS vznikl z potřeby reprezentace komplexnějších datových struktur, než jaké je možné modelovat pomocí hierarchických struktur a z potřeby vytvoření databázového standardu. Zde jsou *„data reprezentována jako kolekce záznamů a relace jedna k více jsou reprezentovány jako množiny.“* (Conolly, 2009, s. 48)
- Relační systém řízení databáze (RDBMS) – Vychází z relačního modelu dat. *„Tento datový model je poměrně jednoduchý, umožňuje vysoký stupeň nezávislosti dat a všechna data reprezentuje ve formě tabulek. Klíčovým vývojovým krokem této doby byl SQL, standartní jazyk pro dotazování v relačních databázích. Označují se jako DBMS druhé generace“* (Conolly, 2009, s. 48)
- Objektově orientované systémy řízení databáze (OODBMS) a objektově relační systémy řízení databáze (ORDBMS) vznikly z důvodu stále se zvyšující složitosti databázových aplikací a jejich požadavků. Oba patří do

DBMS třetí generace. *„Základním principem objektové technologie je zásada, že veškerý software by se měl skládat pokud možno ze standartních, opakovaně použitelných komponent.“* (Conolly, 2009, s. 429)

Složky prostředí DBMS

„Je možné identifikovat pět hlavních složek prostředí DBMS: hardware, software, data, procedury a osoby.

- *Hardware – počítačový systém (systémy), na němž běží databáze. Rozsah může být od jednoho PC k střediskovému počítači nebo počítačové síti.*
- *Software – software DBMS a databázových aplikací, společně s operačním systémem, včetně síťového softwaru, pokud se DBMS používá na síti.*
- *Data – data fungují jako můstek mezi hardwarovou a softwarovou složkou a lidskou složkou. Databáze obsahuje jak provozní data, tak metadata.*
- *Procedury – instrukce a pravidla, která řídí návrh a používání databáze. Mohou obsahovat instrukce o tom, jak se přihlásit k databázi, vytvářet záložní kopie databáze a zvládat selhání softwaru a hardwaru.*
- *Osoby - sem patří návrháři databáze, správci dat (DA), správci databáze (DBA), aplikační programátoři a koncoví uživatelé.“* (Conolly, 2009, s. 41)

2.3.2 Komponenty relační databáze

Komponenty jsou jednotlivé části relační databáze, mezi které patří tabulky, entity, omezení, pohledy a relace. Pojem relace může mít více významů. V tomto případě bude použit ve smyslu vztahu mezi tabulkami.

Tabulka – *„je hlavní jednotkou pro ukládání dat v relační databázi. Tvoří ji dvourozměrná struktura, která se skládá z řádků a sloupců. Každá tabulka představuje entitu, která se má v databázi znázornit.“* (Oppel, 2008, s. 10)

Entita je *„množina objektů se shodnými vlastnostmi, které uživatel identifikuje jako nezávisle existující objekty.“* (Conolly, 2009, s. 156) Například věc nebo činnost.

Omezení – *„je pravidlo aplikované na databázový objekt (většinou tabulku nebo sloupec), které nějakým způsobem omezuje přípustné hodnoty dat tohoto databázového objektu. Jakmile se jednou omezení nastaví DBMS je automaticky vynucuje a nelze je obejít.“* (Oppel, 2008, s. 13)

- **NOT NULL** – toto omezení znemožní použití hodnot null pro daný sloupec v tabulce. Hodnota null se nerovná ničemu, ani jiné hodnotě null. *„Je to speciální kód, který v databázi nemá žádný jiný význam. Informuje o tom, že hodnota sloupce v daném řádku není známa. Hodnota null neodpovídá chybějícímu údaji, prázdnému řetězci ani nule.“* (Oppel, 2008, s. 56)
- **Omezení primárního klíče** zajistí jedinečnost hodnot primárního klíče v rámci tabulky. Lze ho definovat pouze na sloupci, který má definováno omezení not null. Podrobněji je primární klíč vysvětlen v kapitole 2.3.3. Normalizace.
- **Omezení jedinečnosti** - pokud je definováno na sloupci, musí tento sloupec obsahovat jedinečné hodnoty, žádné se nesmí opakovat. Na rozdíl od omezení primárního klíče může tento sloupec obsahovat hodnoty null.
- **Referenční omezení** *„vynucuje relaci mezi dvěma tabulkami v relační databázi. Tzn. RDBMS automaticky kontroluje, zda pro všechny hodnoty cizího klíče existuje odpovídající hodnota primárního klíče v nadřízené tabulce.“* (Oppel, 2008, s. 14)
- **Omezení CHECK** – *„Ověřuje hodnotu sloupce pomocí jednoduchého logického příkazu (v jazyce SQL). Výsledkem příkazu musí být logická hodnota true nebo false. Výsledek true přitom umožní vložit hodnotu sloupce do tabulky, zatímco při výsledku false je hodnota sloupce odmítnuta s odpovídající chybovou zprávou.“* (Oppel, 2008, s. 14)

Pohledy *„poskytují uživatelům databáze mnoho výhod, protože dovolují přizpůsobit data individuálním požadavkům a zpřehledňují zobrazení. Pohled je v zásadě uložený dotaz SQL, na který se lze dotazovat stejně, jako by jednalo o skutečnou tabulku. Pohledy lze přirovnat k virtuálním tabulkám, protože se chovají jako tabulky i když fyzicky ve formě tabulek neexistují.“* (Oppel, 2008, s. 63)

Relace je *„Množina smysluplných spojení mezi entitami.“* (Conolly, 2009, s. 157)

„Relace představují souvislosti mezi tabulkami relační databáze. Zatímco každá relační tabulka může existovat samostatně, databáze jsou především o ukládání souvisejících dat. Pomocí relací lze provázat související tabulky formálním způsobem, který je snadno použitelný, chcete-li ve stejném dotazu spojit data z více tabulek, ale flexibilně přitom zahrnout pouze informace, které vás zajímají. To umožnuje přizpůsobit informace v databázi konkrétním potřebám jednotlivce nebo aplikace, kteří budou databázi využívat.“ (Oppel, 2008, s. 11)

„Relace mezi entitami jsou reprezentovány pomocí mechanizmu primárního a cizího klíče. Při rozhodování zda umístit atribut cizího klíče musíme nejprve identifikovat „rodičovskou“ a „dceřinou“ entitu, kterých se relace týká. Rodičovská entita posílá kopii svého primárního klíče do tabulky, která reprezentuje dceřinou entitu, kde tato kopie funguje jako cizí klíč.“ (Conolly, 2009, s. 233)

V databázích je možné vytvořit tři druhy relací. Jedna k jedné (1:1), jedna k více (1:N) a více k více (N:M). V relační databázi, která bude v této práci použita, lze vytvořit vazby jedna k jedné a jedna k více. Proto budou dále vysvětleny pouze tyto dvě.

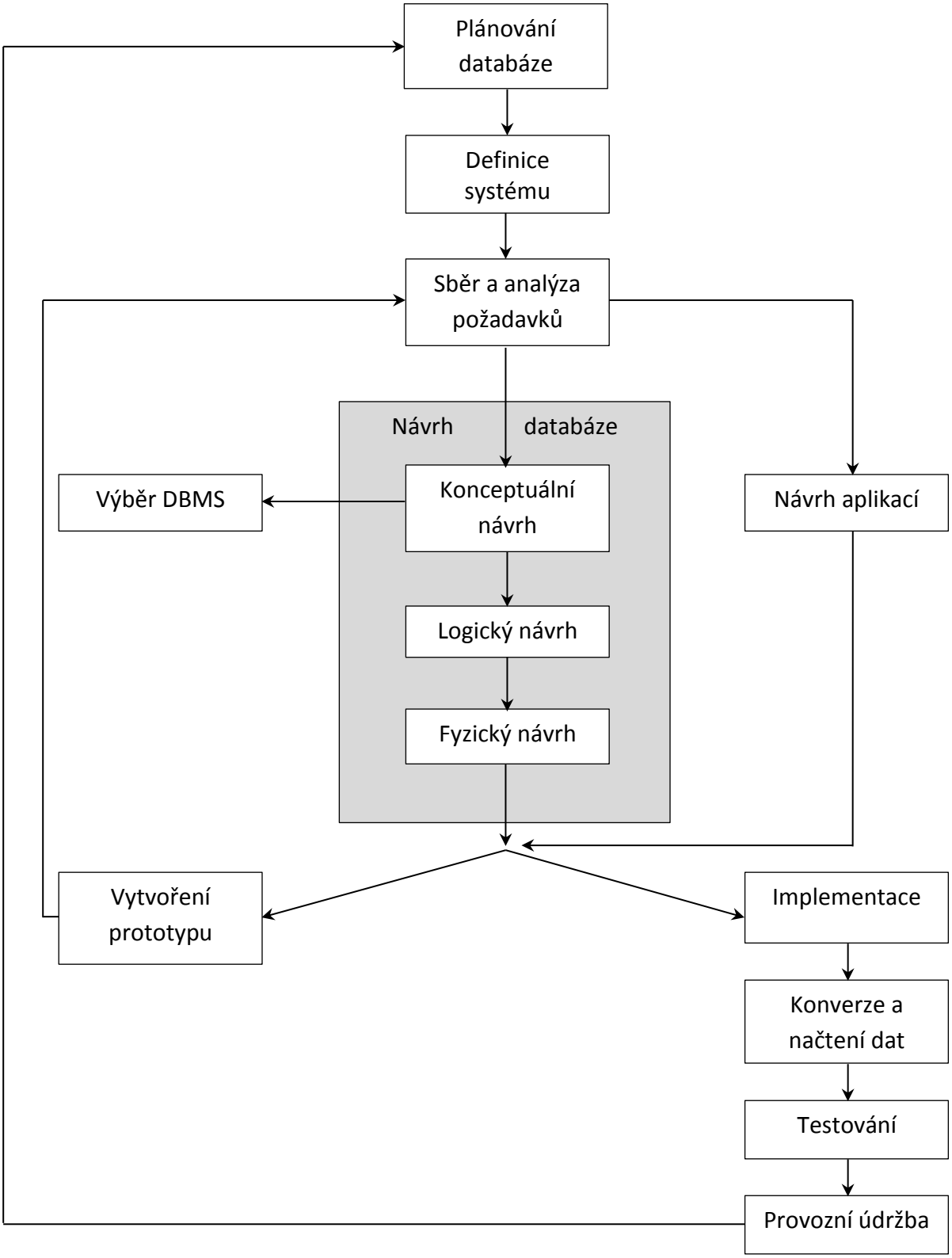
Relaci 1:1 je možné vytvořit, když právě jedna entita v tabulce odpovídá jedné entitě v jiné tabulce. Např. mobilní telefon a jeho výrobní číslo. Když právě jedna entita v tabulce odpovídá jedné nebo více entitám v jiné tabulce, je možno vytvořit relaci 1:N.

2.3.3 Fáze životního cyklu vývoje databázového systému

Tyto fáze životního cyklu popisují postup, který je potřeba dodržet při vývoji databázového systému. Jednotlivé kroky nemusí být uskutečněny v přesném pořadí. Pokud vznikne při vývoji problém, je dobré využít zpětné vazby a některý krok zopakovat nebo úplně přeskočit. Nejlepší možností jak cyklus zobrazit je využít vývojový diagram (Obrázek_4). Jak je vidět, důležitou částí vývoje databázového systému je návrh databáze, ve kterém se mimo jiné vytvoří ER diagram (zobrazuje entity a relace mezi nimi) a popis tabulek. Je rozdělen na tři dílčí části: konceptuální návrh, logický návrh a fyzický návrh.

Nejprve je potřeba, v rámci konceptuálního návrhu, identifikovat důležité objekty a relace mezi nimi. Dále v logickém návrhu reprezentovat tyto objekty a relace množinou tabulek. Ve fyzickém návrhu probíhá implementace v DBMS. (Conolly, 2009)

Návrh aplikací pro zpracování dat z databáze probíhá souběžně s vývojem databáze. Aby se předešlo případným pozdějším problémům, např. při získávání dat, je důležité si již v této fázi uvědomit některé aspekty a požadavky aplikace a databázi jim přizpůsobit.



Obrázek_4. Fáze životního cyklu vývoje databázového systému (Conolly, 2009, s. 110)

2.3.4 Normalizace

Normalizace upravuje strukturu entit a relací na strukturu uspořádání tabulek a relací v databázi. Cílem normalizace je zamezit problémům při aktualizaci a odstraňování dat a vytvoření přehlednější a výkonnější databáze.

Základem normalizace je vytvoření jedinečného identifikátoru, který jednoznačně identifikuje každý záznam v tabulce. Může být vytvořen jedním nebo kombinací několika řádků tabulky. Nazývá se primární klíč a vybírá se z kandidátních klíčů.

Kandidátní klíč je *„sloupec nebo množina sloupců, které jedinečně identifikují záznam v relaci.*

Tento klíč pro tabulku má dvě vlastnosti:

- *Jedinečnost – v každém záznamu určuje hodnota kandidátního klíče výlučně daný záznam.*
- *Neredukovatelnost – žádná vlastní podmnožina kandidátního klíče nezajišťuje jedinečné určení záznamů.*

***Primární klíč** je tedy kandidátní klíč, který je vybrán, aby jedinečně určoval záznamy v tabulce.“* (Conolly, 2009, s. 66)

V dalším kroku normalizace je potřeba odstranit vícehodnotové atributy, tzn., že na průsečíku řádku a sloupce musí být pouze jedna hodnota. Takové relaci se říká, že je v **první normální formě**. Aby byla relace ve **druhé normální formě**, musíme odstranit částečné závislosti atributů, tzn. zajistíme, aby všechny atributy byly plně závislé na primárním klíči. Pokud odstraníme i tranzitivní závislosti mezi dvěma atributy a primárním klíčem (tzn. žádný atribut, který není primárním klíčem, není tranzitivně závislý na žádném jiném klíči), bude relace v **třetí normální formě**. (Conolly, 2009)

*„Poslední prakticky užívanou formou je tzv. **Boyce-Coddova normální forma** (BCNF). Tabulka splňuje BCNF, právě když pro dvě množiny atributů A a B platí: A→B a současně B není podmnožinou A, pak množina A obsahuje primární klíč tabulky. Tato forma zjednodušuje práci s tabulkami. Ve většině případů, pokud dobře postupujeme při tvorbě tabulek, aby splňovaly postupně 1NF, 2NF a 3NF, forma BCNF je splněna.“* (Interval.cz, 2000)

2.4 Strukturovaný dotazovací jazyk (SQL)

„SQL je základním jazykem, který slouží ke komunikaci s relačními databázemi. Často se říká, že matematika je jazykem vědy. Obdobně platí, že SQL je jazykem databází.“ (Oppel, 2008, s. 7)

„Jazyk SQL připomíná klasický přirozený jazyk (samozřejmě anglický), má však přesně definovaná syntaktická a lexikální pravidla.“ (Kiszka, 2009, s. 64)

Tento jazyk patří mezi neprocedurální jazyky, takže stačí uvést, která data je nutno získat, ale již není potřeba určovat jakým způsobem. To zařídí SQL modul v databázovém systému. Často je používán v kombinaci s objektově orientovanými nebo procedurálními jazyky. Stejně tomu bude i v této práci, kde bude použit pro získání požadovaných dat z databáze, ale pro zobrazení a prezentaci těchto dat v prohlížeči poslouží jazyk Java.

SQL příkazy se dělí do několika kategorií podle úkonů, který daný příkaz provádí. Skládají se z předem definovaných a z uživatelsky volitelných slov. Předem definovaná, tzv. rezervovaná slova, tvoří jednoduchá anglická slovíčka a je zvykem je psát velkým písmem. Jelikož SQL nerozlišuje velikost písmen, není to vyžadováno. Uživatelsky definovaná slova musí splnit syntaktická pravidla a jsou to názvy databázových objektů např. sloupců a tabulek. (Oppel, 2008)

Dále zde budou jednotlivé kategorie podrobněji rozepsány i s příkladem obecného použití příkazu a podrobnějšími informacemi.

2.4.1 Kategorie příkazů jazyka SQL

DDL (Data Definition Language) – Zde patří příkazy CREATE, ALTER a DROP, které umožňují vytvářet, upravovat a mazat objekty databáze. Tabulky, indexy, pohledy, apod.

Příkaz CREATE vytvoří nový databázový objekt. Podle použité syntaxe se rozlišuje CREATE DATABASE, CREATE TABLE, CREATE INDEX a CREATE VIEW.

Příkaz CREATE DATABASE vytvoří novou databázi.

```
CREATE DATABASE název_databáze [specifické_parametry_dodavatele]
```

Příkaz CREATE TABLE *„patří k nejdůležitějším příkazům jazyka SQL. Relační paradigma vyžaduje, že všechna uložená data musí být umístěna v tabulkách. Chcete-li tedy uložit data do databáze, musíte vždy nejdříve vytvořit tabulku.“* (Oppel, 2008, s. 58)

```
CREATE TABLE název_tabulky(
<definice_sloupce>
[, <definice_sloupce>…]
[, <omezení_tabulky>…] );
```

Příkaz CREATE INDEX vytvoří nový index. Klíčové slovo UNIQUE určuje, zda jde o jedinečný index. ASC, DESC určuje vzestupné nebo sestupné pořadí.

```
CREATE [UNIQUE] INDEX název_indexu ON název_tabulky
(název_sloupce [ASC | DESC] [, název_sloupce [ASC | DESC]…] );
```

Příkaz CREATE VIEW vytvoří nový pohled.

```
CREATE [OR REPLACE] VIEW název_pohledu  AS dotaz_sql;
```

Příkaz ALTER umožňuje měnit již existující databázový objekt. ALTER TABLE umožní změnu téměř všech parametrů nadefinovaných pomoci CREATE TABLE a to většinou bez nutnosti vytvářet tabulku znovu.

Příkaz DROP smaže existující databázový objekt.

```
DROP <typ_objektu> název_objektu [<možnosti_ostranění>];
```

DML (Data Manipulation Language) – Tyto příkazy umožňují načítat, přidávat, měnit a mazat data v databázi. Jsou to SELECT, INSERT, UPDATE a DELETE.

Příkaz SELECT je nejpoužívanější příkaz jazyka SQL. *„Pomocí tohoto příkazu lze načítat data z databáze. Data je pak možné zpracovat v aplikaci nebo je pouze zobrazit uživateli. Odpověď, která se nazývá sada výsledků, je vždy vrácena ve tvaru tabulky.“* (Oppel, 2008, s. 69)

K tomuto příkazu se vážou další klauzule (klíčová slova), která se v něm mohou vyskytnout. Kromě klauzule FROM ostatní nejsou povinné. Pokud však jsou uvedeny, musí být napsány přesně v tomto pořadí:

- FROM uvádí ze které tabulky nebo pohledu budou data získána
- WHERE umožňuje přidat podmínku pro výsledky. Budou zobrazeny pouze výsledky, u kterých se podmínka vyhodnotí logickou hodnotou true.

Podmínka rozsahu (BETWEEN x AND y), podmínka členství v množině (IN (x, y)), porovnávací podmínka (x=y), podmínka prázdných hodnot (IS NULL nebo IS NOT NULL), podmínka řetězce (název_sloupce LIKE 'řetězec')

- GROUP BY zajistí seskupení výsledků dotazu do skupin
- HAVING umožňuje přidat další podmínku. Většinou se používá v kombinaci s klauzulí GROUP BY, kdy se aplikuje na skupiny řádků
- ORDER BY umožní seřadit výsledky dotazu podle jednoho nebo více sloupců

Následující příkaz zobrazí všechna data z tabulky (nepovinné klíčové slovo DISTINCT vyloučí duplicitní řádky):

```
SELECT [DISTINCT] *
FROM název_tabulky;
```

Příkaz INSERT umožňuje přidávat řádky do tabulky. Vkládání probíhá pomocí klauzule VALUES. Každý příkaz umožní vložit pouze jeden řádek.

```
INSERT INTO název_tabulky
[(seznam_sloupců)]
VALUES (seznam_hodnot);
```

Příkaz UPDATE umožňuje upravovat data v jednom nebo ve více sloupcích tabulky. Používá se ve spojení s klauzulí WHERE, kde lze přesně specifikovat, o které řádky se jedná. Pokud není uvedena, upraví všechny řádky v tabulce.

```
UPDATE název_tabulky
SET název_slouce = výraz
    [, název_sloupce = výraz…]
[WHERE Klauzule];
```

Příkaz DELETE smaže jeden nebo více řádků tabulky. Aby bylo jasné, o které řádky se jedná, opět se používá ve spojení s klauzulí WHERE. Pokud není uvedena, smaže všechny řádky v tabulce.

```
DELETE FROM název_tabulky
[WHERE Klauzule];
```

DCL (Data Control Language) – Slouží správcům databáze. Pomocí klíčových slov SHUTDOWN, BACKUP DATABASE, GRANT a REVOKE, umožňuje spravovat oprávnění a řídit přístup k datům. Například spouštět a vypínat databázi.

SHUTDOWN umožňuje vypnout server.

BACKUP DATABASE umožnuje zálohovat databázi SQL server.

Příkaz GRAND umožnuje přidělovat uživatelům databáze oprávnění.

```
GRAND oprávnění [, oprávnění…]
[ON objekt]
TO příjemce [, příjemce…];
[WITH GRAND OPTION | WITH ADMIN OPTION]
```

Příkaz REVOKE umožnuje odebírat oprávnění uživatelům databáze.

```
REVOKE oprávnění [, oprávnění…]
[ON objekt]
FROM příjemce [, příjemce…];
```

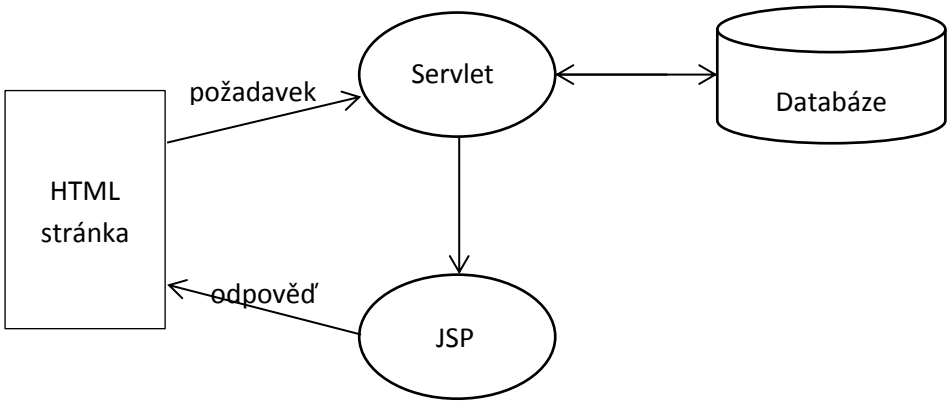
2.5 Java Servlet a JavaServer Pages

Problematika programovacího jazyka Java je složitá a velmi obsáhlá. Proto zde budou zmíněny pouze nejdůležitější informace týkající se základních principů jeho funkce a tvorby kódu.

Jak je již zmíněno v předchozí kapitole, jazyk Java bude v této práci použit pro prezentaci dat v prohlížeči. Konkrétně Java Servlet a JSP. Tato technologie slouží k tvorbě dynamických webových stránek a webových aplikací. Fungování Java Servletu a JavaServer Pages je zjednodušeně zobrazeno na následujícím obrázku (obrázek_5).

Pro obsluhu požadavků ze strany klienta je využíváno různých metod uložených v těle Java Servletu, který tvoří jazyk Java. Ten se ale příliš nehodí pro psaní webových stránek, protože při generování každého řádku html kódu musí být volána metoda „out.print“. To je ale velice nepohodlné, zdlouhavé a nepřehledné a proto vznikly JavaServer Pages, které jsou z velké části tvořeny přímo html kódem a Java je použita pouze v nejnutnějších případech.

Klient vyšle požadavek. Java Servlet získá z hlavičky a z těla tohoto požadavku příslušné parametry a zpracuje požadavek, popřípadě získá (upraví nebo smaže) data z databáze. K tomu využívá metody napsané v těle Servletu. Dále je vygenerována JSP a poslána klientovi.



Obrázek_5. Zjednodušený pohled na technologii Java Servlet a JSP (Zdroj: vlastní)

2.5.1 Formuláře JSP stránek

Pro získání potřebných dat od uživatele aplikace se v JSP využívá formulářů. Formulář se vytvoří v těle JSP použitím značky **<FORM>**. U této značky je potřeba definovat parametry NAME, METHOD a ACTION.

- NAME je název formuláře. Musí být pojmenován, aby bylo možné se na něho odkazovat z dalších komponent.
- METHOD určuje metodu, jaká bude použita pro komunikaci pomocí protokolu http. Použity mohou být metody GET nebo POST, přičemž při generování dynamických stránek je používanější POST. U metody GET se pro přenos parametrů využívá části URL. To zahrnuje hned dvě nevýhody. URL má omezenou délku a nelze použít pro skryté zadání parametru požadavku. Metoda POST používá pro uložení parametrů tělo dotazu, takže není omezena jejich délka.
- ACTION je poslední povinný parametr u značky FORM. Určuje jaká akce má být provedena na straně serveru při přijetí požadavku ze strany klienta.
- Parametr ENCRYPTE, který udává typ odesílaných dat, se nemusí definovat, pokud se ze strany klienta odesílá pouze text.

Další značky, které se používají uvnitř formulářů, umožňují vytvořit různé ovládací tlačítka, textová pole, přepínače, atd. Značka **<INPUT>** má nejdůležitější parametry TYPE, NAME a VALUE.

- TYPE, určuje typ ovládacího prvku. Jsou to TEXT (textové pole), PASSWORD (textové pole pro heslo, automaticky skrývá znaky), HIDDEN (skrytý prvek), SUBMIT (tlačítko pro odeslání formuláře), RESET (tlačítko pro resetování formuláře, BUTTON (tlačítko), CHECKBOX (zaškrtávací políčko) a RADIO (přepínač).
- NAME je název, pod kterým bude uložen parametr z příslušného ovládacího prvku při odeslání formuláře.
- VALUE je hodnota parametru, která se při odeslání formuláře uloží pod jménem NAME.

Značka **<BUTTON>** vytvoří v podstatě stejné tlačítko jako v případě použití značky <INPUT> s parametrem SUBMIT, RESET nebo BUTTON s tím rozdílem, že hodnota této značky není hodnotou parametru pro stranu klienta. **<SELECT>** umožňuje vytvořit rozbalovací výběr nebo výběr s posunovacím seznamem.

2.5.2 JSTL značky

Dříve se hojně používaly scriptlety, což je kód java vložený do JSP stránky. Začátek a konec je označen znaky <% a %>. Nicméně nevýhodou tohoto řešení je nepřehlednost a složitá úprava těchto vložených kódů. Proto se pro vytvoření přehlednějšího kódu JSP stránek využívá JSTL značek a EL neboli JSP Standard Tag Library a Expression Language.

Před použitím těchto značek je nutné deklarovat knihovnu značek na začátku JSP stránky. Uvádí se prefix, který určuje použitou zkratku a URI adresa knihovny těchto značek. Deklarace vypadá takto:

```
<%@ taglib prefix="zkratka" uri="adresa_knihovny_značek" %>
```

Dále jsou specifikovány jednotlivé značky použité v této práci.

JSTL značky s prefixem „c“:

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/fmt" %>

- c:out - zobrazí výsledek výrazu podobně jako <%= výraz%>.
- c:set – vyhodnotí výraz a výsledky umožní uložit do atributu.
- c:remove – odstraní hodnotu libovolného parametru.
- c:catch – odchytí jakoukoliv výjimku, která se vyskytne v těle JSP.
- c:if – vyhodnotí výraz a zobrazí co je uvnitř pouze pokud je výsledek true.
- c:choose – umožňuje vybírat mezi alternativami c:when a c:otherwise.
- c:when – používá se uvnitř značky c:choose a obsahuje nějakou podmínku.
- c:otherwise - používá se uvnitř značky c:choose. Zobrazí obsah pokud c:when je false.
- c:import – poskytuje všechny funkce <include>. Také může obsahovat absolutní URL.
- c:forEach – iteruje přes kolekce objektů. Vytvoří cyklus.
- c:param – umožnuje nastavit jako hodnotu parametru URL.
- c:redirect – přesměruje prohlížeč na alternativní URL adresu.
- c:url – v případě potřeby přepíše URL. Poskytuje vždy správné kódování. (tutorialspoint.com, 2013)

JSTL značky s prefixem „sql“:

<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>

- sql:setDataSource – konfigurační proměnná. Nastaví zdroj dat.
- sql:query – vyvolá jakýkoliv příkaz SQL SELECT a výsledek uloží.
- sql:update – vyvolá příkaz SQL, který nevrací data (INSERT, UPDATE, DELETE).
- sql:param – používá se v sql:query a sql:update. Vloží hodnotu proměnné.
- sql:dateParam – používá se v sql:query a sql:update. Ukládá hodnotu data a času.
- sql:transaction – používá se pro hromadné transakce sql:query a sql:update. (tutorialspoint.com, 2013)

Expression Language

Pro správnou funkci stránek nebo aplikaci je nezbytné používat a ukládat atributy stránky, requestu, session nebo aplikace. A právě k těmto atributům EL usnadňuje přístup. Výraz expression language je označen znaky ${}, ve kterých bývá většinou název příslušného parametru, který je potřeba zobrazit. Tento výraz může být složen také z operátorů:

- aritmetických (+, -, *, /, div, %, mod),
- logických (and, &&, or, ||, not, !),
- relačních (==, eq, !=, ne, <, lt, >, gt, <= , ge, >=, le),
- podmíněných operátorů (podmínka ? splněno : nesplněno),
- logického operátoru „empty“.

2.6 Použitý software

Při konzultaci ve firmě Xevos Solutions s.r.o. byly dohodnuty požadavky na informační systém vytvářený v této práci. Také bylo dohodnuto, jaký software bude použit.

Shrnutí dohodnutých požadavků firmy

- přehledná a intuitivní webová aplikace, která zobrazí známky žáků a sdělení učitelů rodičům
- vkládání, úprava a mazání výše zmíněných dat přes webovou aplikaci
- databáze bude implementována v MS SQL Serveru
- bude zajištěna určitá ochrana dat před neoprávněným zobrazením nebo úpravou

Pro vytvoření webové aplikace bude použit program NetBeans IDE se serverem GlassFish. Databáze bude implementována v Microsoft SQL Server Express Edition.

NetBeans IDE

„NetBeans je úspěšný Open Source projekt s velmi rozsáhlou uživatelskou základnou, rostoucí komunitou vývojářů a téměř 100 partnery po celém světě. Firma Sun Microsystems založila Open Source projekt NetBeans v červnu 2000 a je zároveň hlavním sponzorem celého projektu.

Vývojové prostředí NetBeans IDE je nástroj, pomocí kterého programátoři mohou psát, překládat, ladit a distribuovat aplikace. Samotné vývojové prostředí je vytvářeno v jazyce Java - ovšem podporuje prakticky jakýkoliv programovací jazyk. Existuje rovněž velké množství modulů, které toto vývojové prostředí rozšiřují. Vývojové prostředí NetBeans je bezplatně šířený produkt a jeho užívání není nijak omezeno.“ (netbeans.org, 2013)

GlassFish

Glassfish je open source Java EE aplikační server vyvinutý opět firmou Sun Microsystems. Je to software, který umožnuje aplikacím poskytovat služby přes internet. (Oracle.com, 2013)

Podporuje Enteprise JavaBeans, servlety, JavaServer Pages, JavaServer Faces atp. Slouží především jako prostředí pro nasazení a spouštění podnikových aplikací v internetové síti.

Hlavními výhodami jsou:

- web kontejner,
- lehká správa a konfigurace,
- dynamická reakce na aktuální zátěž,
- centralizovaná správa životního cyklu aplikací,
- nástroje pro aktualizaci a přídavné moduly. (java.net, 2013)

Microsoft SQL Server Express Edition

Microsoft SQL Server je relační databázový systém sloužící pro správu většího množství dat ať už přes podnikovou síť, na lokálním počítači nebo přes internet.

„SQL Server Express je bezplatná edice systému SQL Server, ideální pro vývoj a nasazení aplikací pro klientské počítače, web a malé servery. Hlavní funkce a možnosti využití:

- *Podpora 10 GB úložného místa na databázi*
- *Snadné zálohování a obnovení*
- *K dispozici s grafickým nástrojem pro správu*
- *Rychlé vytvoření a nasazení databázového řešení*
- *Vytváření webů a webových aplikací*
- *Vytváření malých obchodních aplikací“* (microsoft.com, 2013)

3 Návrh IS a jeho realizace

Jak již bylo zmíněno v teoretické části této práce, tvorba webové aplikace probíhá souběžně s tvorbou databáze. Proto prvním krokem je vytvoření konceptuálního návrhu databáze s přihlédnutím k nárokům a požadavkům aplikace. Dále následuje logický a fyzický návrh.

3.1 Konceptuální návrh

Tomuto návrhu předchází definování hranic systému a analýza požadavků na systém i na aplikaci, což je shrnuto v diagramu případu užití (Obrázek_6). Tento diagram zobrazuje vztahy systému ke svému okolí (k uživatelům) a v systému samotném. Pro vytvoření korektního diagramu jsou použity dvě tabulky. První (Tabulka_1) popisuje profily aktérů, druhá (Tabulka_2) obsahuje jejich cíle a činnosti potřebné pro splnění požadavků.

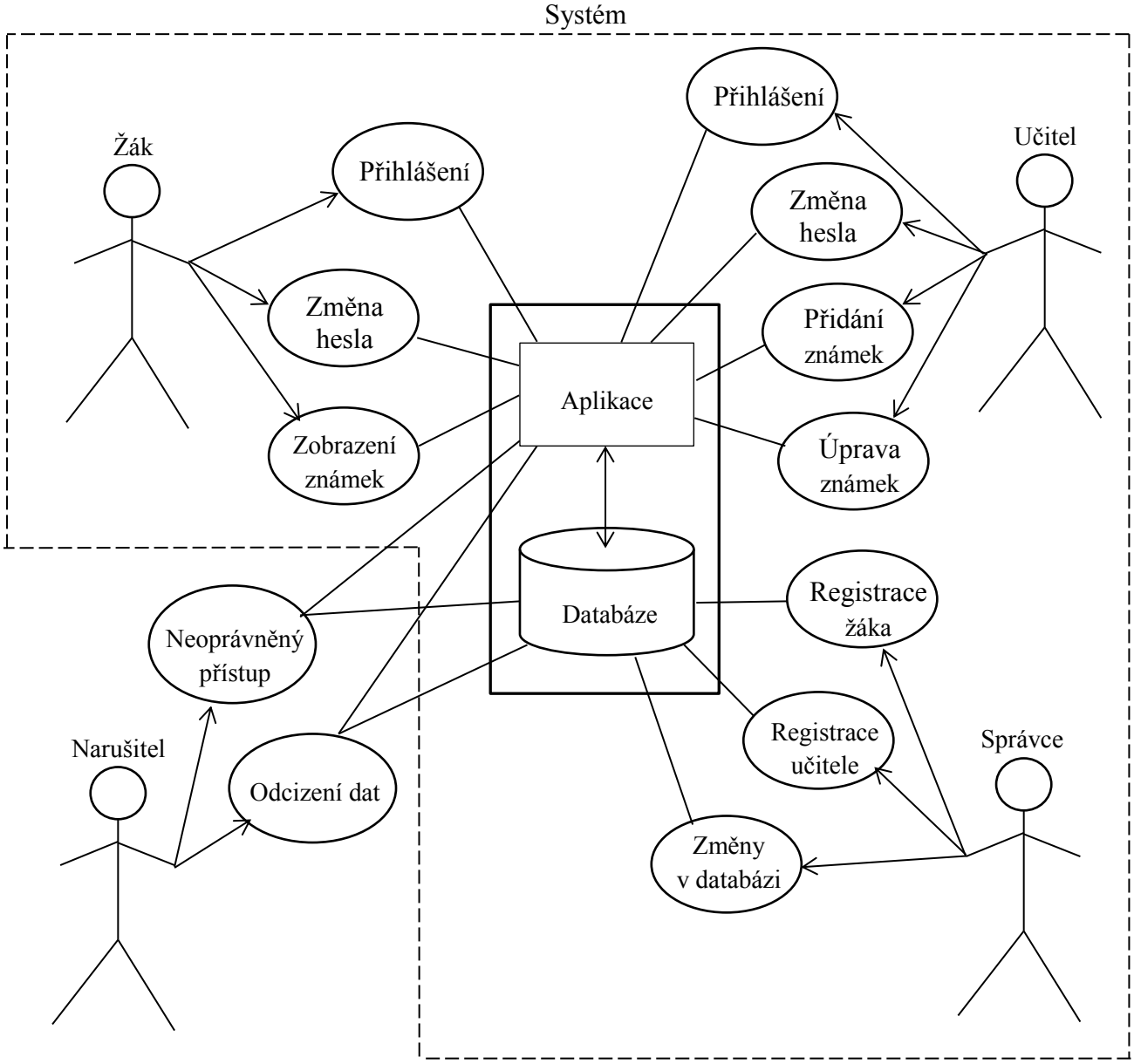
Aktér	Profil: Kvalifikace a schopnosti
Žák	Žák školy (nebo jakýkoliv rodinný příslušník), který chce zobrazit své výsledky ze školy a vlastní přihlašovací jméno a heslo potřebné pro přihlášení do aplikace.
Učitel	Osoba, která vyučuje žáka nebo je pověřena udělovat žákovi hodnocení a známky do databáze. Je poučena o funkci aplikace.
Správce	Osoba, pověřená správou databáze (přidáváním a upravováním veškerého obsahu). Má přímý přístup do databáze a je poučena a znalá funkce systému.
Narušitel	Jakákoliv osoba, která má zájem nebo se snaží o neoprávněný přístup k aplikaci nebo do databáze.

Tabulka_1. Profily aktérů (Zdroj: vlastní)

Aktér	Požadavek (cíl)	Činnost
Žák	Informace o prospěchu	Přihlášení do aplikace
Učitel	Informovat o prospěchu žáky a rodinné příslušníky	Přihlášení do aplikace Přidání a aktualizace známek
Správce	Udržovat databázi aktuální a bezpečnou	Přidání, mazání a úprava dat Řešení problémů
Narušitel	Narušit bezpečnost aplikace a databáze	Zneužití osobních údajů

Tabulka_2. Cíle a záměry aktérů (Zdroj: vlastní)

Nyní, s využitím předchozích tabulek, je možno sestrojit diagram případu užití z pohledu jednotlivých aktérů, který bude dále využit při tvorbě ER diagramu.



Obrázek_6. Diagram případu užití. (Zdroj: vlastní)

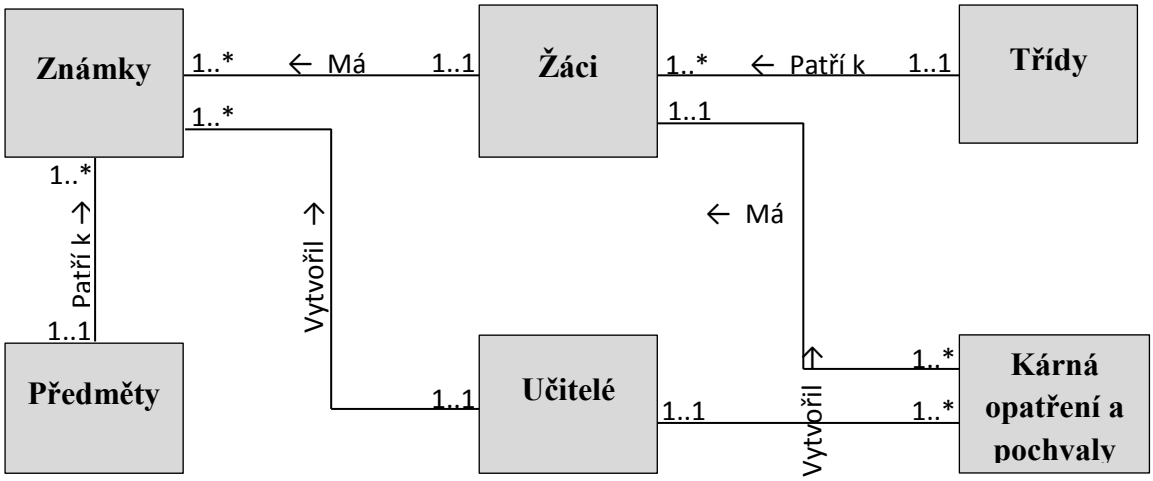
Dalším a zároveň posledním krokem před tvorbou samotného ER diagramu (Obrázek_7) je identifikace entit a jejich vzájemných vztahů. V tabulce jsou zobrazeny názvy a popisy entit (Tabulka_3) a jejich kardinalita vztahů (Tabulka_4).

Název entity	Popis entity
Předměty	Předměty, které se vyučují v dané škole
Třídy	Jednotlivé třídy žáků ve škole
Učitelé	Vyučující ve škole
Žáci	Všichni žáci, kteří navštěvují školu
Známky	Studijní výsledky žáků
KO a P	Kárná opatření a pochvaly žáků

Tabulka_3. Název a popis entit. (Zdroj: vlastní)

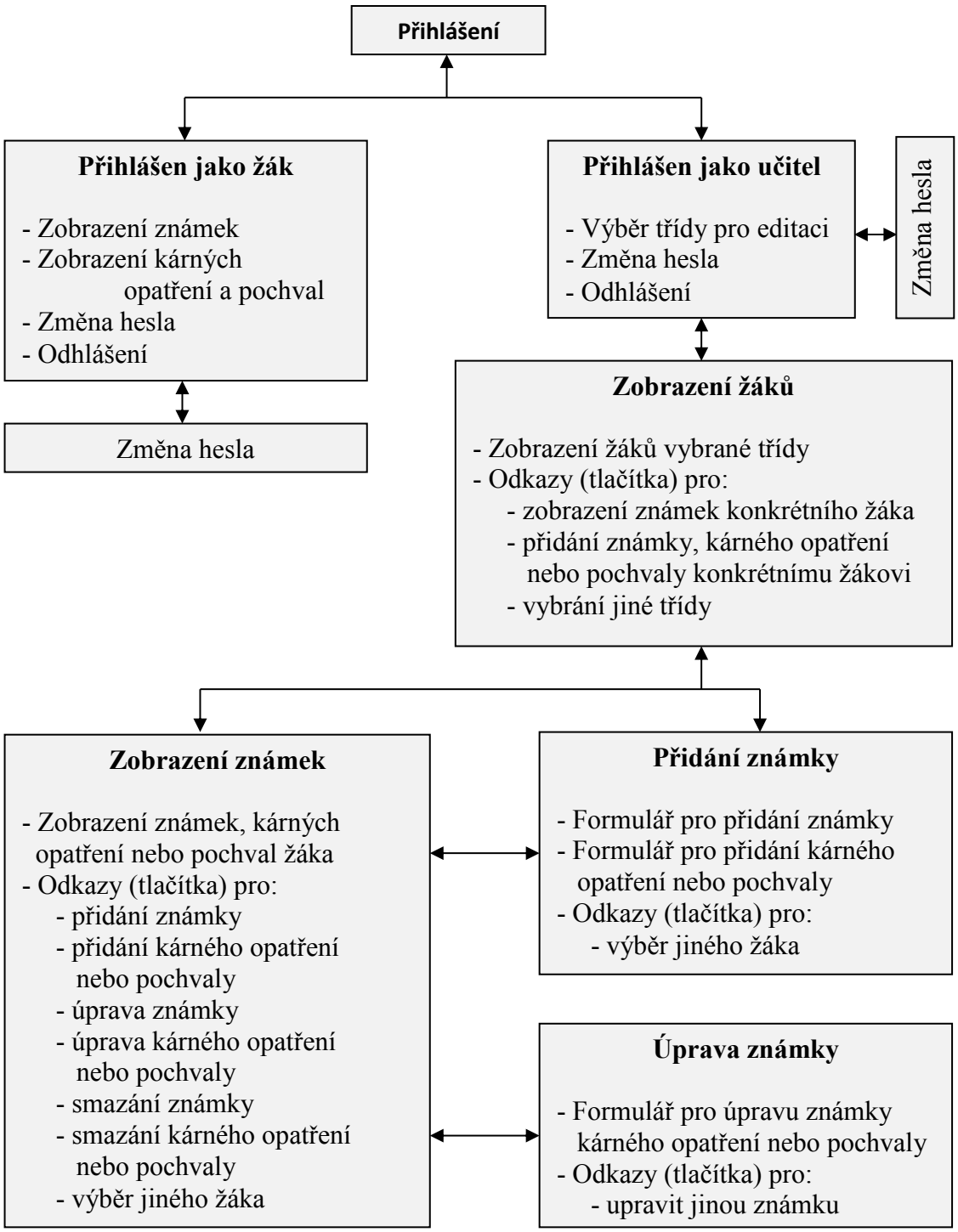
Vztah	Kardinalita vztahu	Popis
Třídy / Žáci	1:N	Třída může mít N žáků ale žák pouze jednu třídu
Žáci / Známky	1:N	Žák může mít N známek, ale známka se váže pouze na jednoho žáka
Žáci / KO a P	1:N	Žák může mít N kárných opatření a pochval, ale KO nebo pochvala se váže pouze na jednoho žáka
Učitelé / Známky	1:N	Učitel může udělit N známek, ale udělená známka se váže pouze k jednomu učiteli
Předměty/Známky	1:N	Předmět může mít N známek ale známka pouze jeden přiřazený předmět
Učitelé / KO a P	1:N	Učitel může udělit N kárných op. a pochval ale KO nebo pochvala se váže pouze na jednoho učitele

Tabulka_4. Entity a jejich vztahy. (Zdroj: vlastní)



Obrázek_7. ER diagram (Zdroj: vlastní)

Nyní je vytvořen diagram konceptuálního návrhu aplikace (Obrázek_8), který zobrazuje jednotlivé stránky webové aplikace s výčtem činností, provázaností jednotlivých stránek a s dalšími prvky potřebnými pro správné a intuitivní ovládání.



Obrázek_8. Konceptuální návrh aplikace. (Zdroj: vlastní)

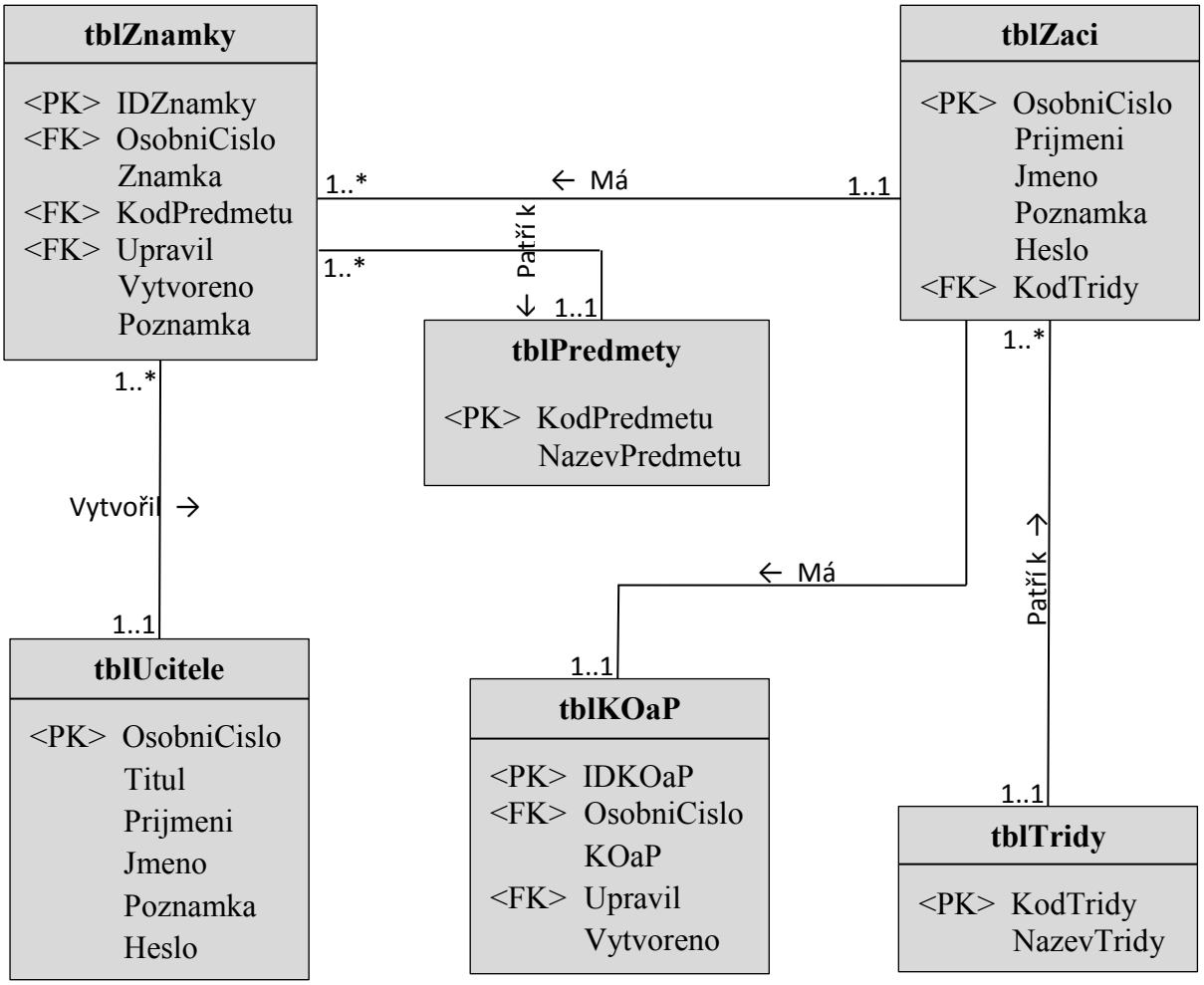
3.2 Logický návrh

V této části je vytvořen návrh tabulek a sloupců databáze včetně datového typu každého sloupce a případného omezení. U tohoto návrhu se vychází z ER diagramu. Nejsou v něm žádné vazby N:M, což značí dobře provedený konceptuální návrh a nemusí se tedy dekomponovat tabulky. Již zde je možné a nutné navrhnout primární a cizí klíče a tím zajistit entitní a referenční integritu dat. Všechno je přehledně zobrazeno v následujícím slovníku dat (Tabulka_5).

Tabulka	Sloupec	Datový typ (délka)	Omezení
tblPredmety	KodPredmetu	int	PK
	NazevPredmetu	nvarchar(30)	not null
tblTridy	KodTridy	tinyint	PK
	NazevTridy	nchar(3)	not null
tblUcitele	OsobniCislo	nchar(7)	PK
	Titul	nchar(4)	
	Prijmeni	nvarchar(30)	not null
	Jmeno	nvarchar(30)	not null
	Poznamka	nvarchar(100)	
	Heslo	nvarchar(20)	not null
tblZaci	OsobniCislo	nchar(7)	PK
	Prijmeni	nvarchar(30)	not null
	Jmeno	nvarchar(30)	not null
	Poznamka	nvarchar(100)	
	Heslo	nvarchar(20)	not null
	KodTridy	tinyint	FK
tblZnamky	IDZnamky	int	PK, Identity (1,1)
	OsobniCislo	nchar(7)	FK
	Znamka	tinyint	not null
	KodPredmetu	int	FK
	Poznamka	nvarchar(100)	
	Upravil	nchar(7)	FK
	Vytvoreno	nchar(16)	not null

tblKOaP	IDKOaP	int	PK, Identity (1,1)
	OsobniCislo	nchar(7)	FK
	KOaP	nvarchar(100)S	Not null
	Upravil	nchar(7)	FK
	Vytvoreno	nchar(16)	not null

Tabulka_5. Slovník dat. (Zdroj: vlastní)



Obrázek_9. Logický návrh databáze (Zdroj: vlastní)

Pro úplné objasnění logického návrhu databáze (Obrázek_9) dále následuje podrobný popis tabulek a sloupců. V tabulkách tblUcitele a tblZaci byl zvolen jako primární klíč sloupec OsobniCislo. Bude v něm zkratka složena z několika písmen ze jména a příjmení a z pořadové číslice učitele nebo žáka, které budou jednoznačně identifikovat každý řádek tabulky. Dále obsahují sloupec Poznamka, kde je místo pro jakýkoliv důležitý dovětek (maximálně 100 znaků), a sloupec Heslo kde bude uloženo přidělené nebo samotnou osobou

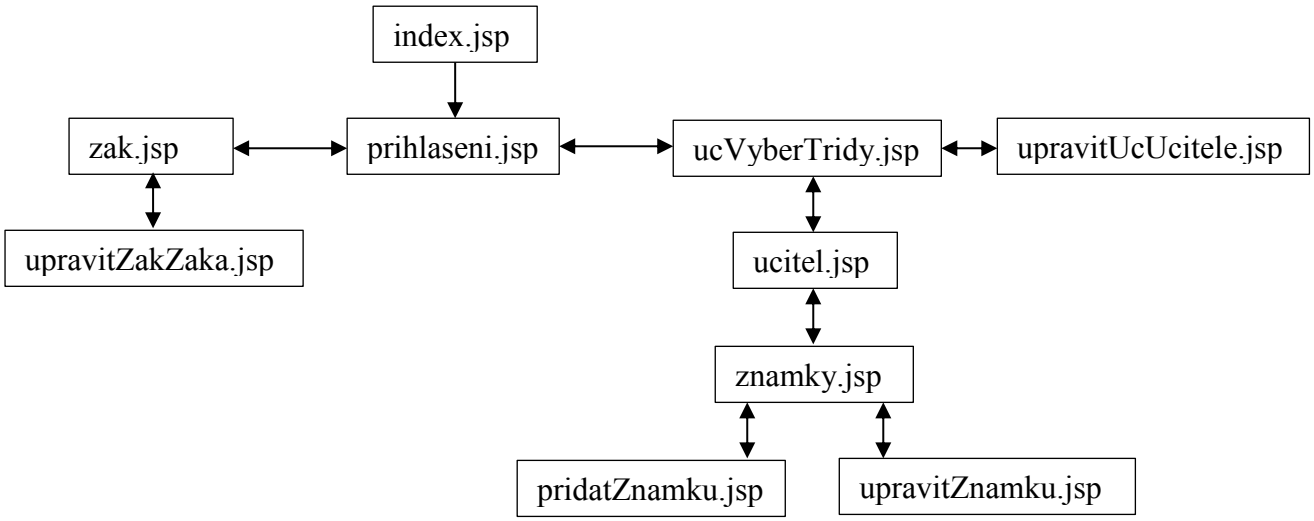
vytvořené heslo. Aby bylo možné žáky řadit nebo zobrazit podle třídy, do které jsou přiřazeni, je v tabulce tblZaci navíc sloupec KodTridy, který je cizím klíčem sloupce KodTridy z tabulky tblTridy.

Dalšími tabulkami jsou tblPredmety a tblTridy. Jak již název napovídá, tyto tabulky slouží pro uložení všech předmětů vyučovaných na škole a všech tříd, do kterých lze přiřadit žáky. Skládají se vždy ze dvou sloupců. KodTridy (číslo od 0 do 255) a KodPredmetu jsou zvoleny jako primární klíče a budou vytvářeny podle daného pravidla. Ostatní dva sloupce slouží pro uložení názvu předmětu a názvu třídy.

Následující tabulkou je tblZnamky. Slouží pro uložení učitelem udělených známek žákovi. Primárním klíčem je zvolen sloupec IDZnamky, který pro to byl speciálně vytvořen. Vyplní se automaticky při přidání záznamu a to vždy číslem o jedno větší než poslední přidaný záznam. Sloupec OsobniCislo je cizím klíčem sloupce OsobniCislo z tabulky tblZaci a je vytvořen pro „svázání“ známky s konkrétním žákem. Dále obsahuje sloupec KodPredmetu, který je cizím klíčem sloupce KodPredmetu z tabulky tblPredmety a slouží pro „svázání“ známky s předmětem, z kterého žák tuto známku obdržel. Do důležitého sloupce Upravil, který je cizím klíčem sloupce OsobniCislo z tabulky tblUcitele, se při vytvoření záznamu automaticky ukládá informace o tom, kým byla známka vytvořena nebo naposledy upravena. Zároveň se automaticky do sloupce Vytvoreno uloží aktuální datum a čas úpravy. Samozřejmě ani zde nechybí sloupec pro napsání poznámky, kterou uvidí i žák při výpisu známek v aplikaci.

Poslední tabulka je nazvána tblKOaP jako zkratka kárné opatření a pochvaly. Obsahuje sloupec IDKOaPO, který je primárním klíčem a ukládá se do něj automaticky číslo záznamu. Sloupec OsobniCislo slouží pro uložení osobního čísla žáka, kterému patří to které kárné opatření nebo pochvala. KOaP je sloupec s datovým typem text a slouží pro uložení samotného kárného opatření nebo pochvaly. Sloupce Upravil a Vytvoreno mají stejnou funkci jako v tabulce tblZnamky a to uložení data a času vytvoření nebo poslední úpravy záznamu a uložení osobního čísla učitele, aby bylo jasné, kdo a kdy záznam vytvořil nebo upravil.

Dále je vytvořen diagram logického návrhu aplikace (Obrázek_10), který znázorňuje provázanost jednotlivých stránek aplikace a také jejich jména přesně ve tvaru v jakém budou reálně vytvořeny.



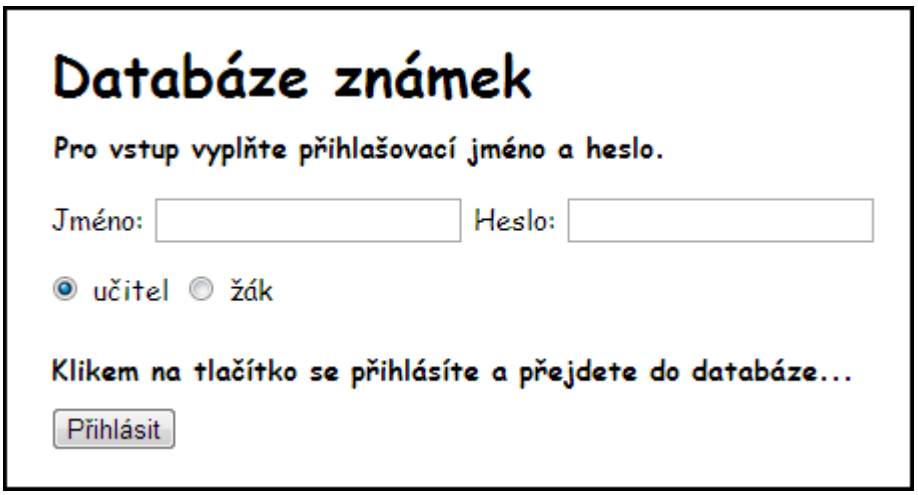
Obrázek_10. Logický návrh aplikace. (Zdroj: vlastní)

3.3 Fyzický návrh

V této části bude podle logického návrhu vytvořena funkční databáze v Microsoft SQL Server a webová aplikace s využitím technologie Java Servlet a JSP. Budou zde také popsány základní funkční elementy, které je nutno vytvořit pro spolupráci těchto dvou součástí informačního systému.

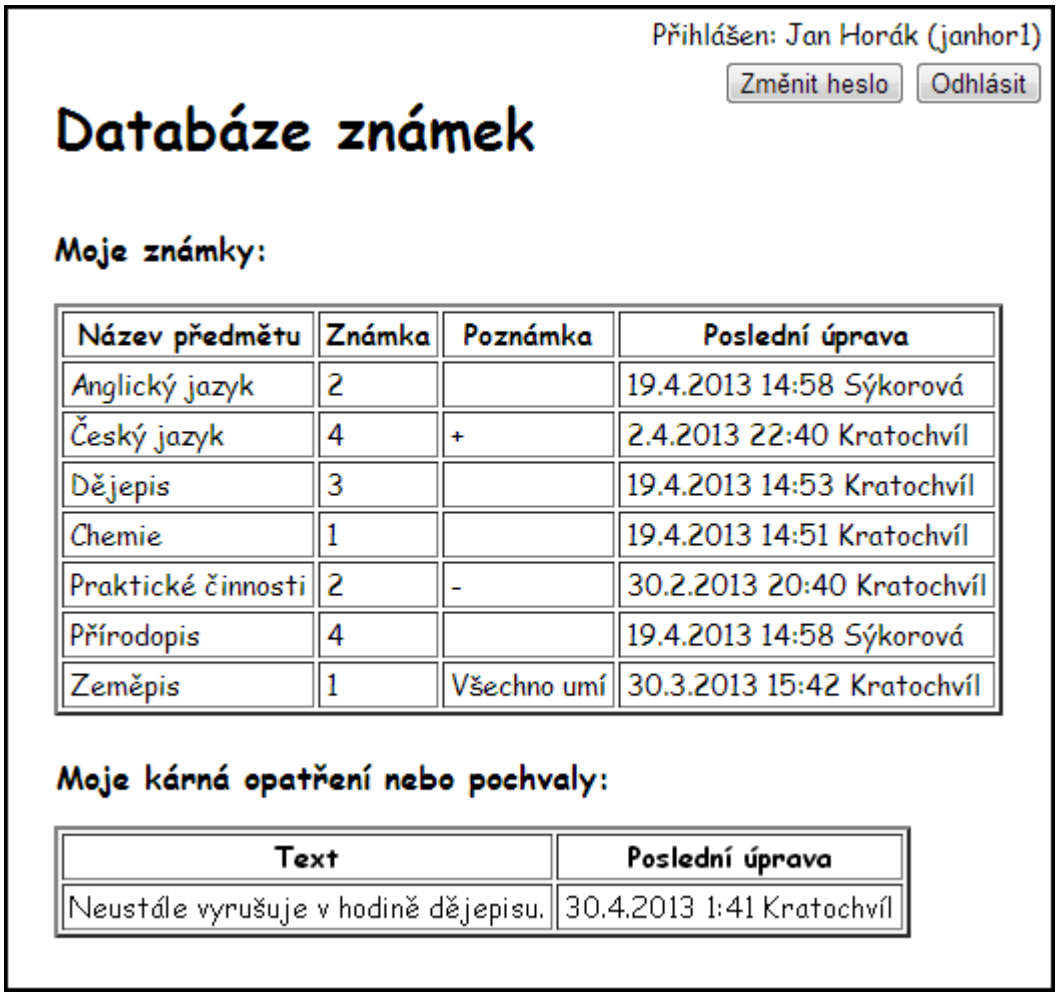
Prvním krokem je tedy vytvoření databáze. Databázi tvoří dva soubory. Data file (s příponou .mdf) a Log file (přípona .ldf). Oba tyto soubory jsou, po upřesnění názvu a velikosti databáze, vytvořeny v programu Microsoft SQL Server Management Studio. Nyní se podle slovníku dat vytvoří všechny tabulky a sloupce. Nastaveny jsou datové typy i omezení sloupců včetně primárních a cizích klíčů. Dále je nutno tabulky naplnit testovacími daty aby se odhalily případné chyby a nedostatky, které je potřeba odstranit. Pomocí jazyka SQL jsou napsány a uloženy procedury nutné pro přidávání nebo úpravu dat přes webovou aplikaci.

V konceptuálním návrhu, můžeme vidět, že žák i učitel se musí pro vstup do informačního systému přihlásit svým uživatelským jménem a heslem (Obrázek_11). Je to jak z důvodu bezpečnosti, tak pro odlišení přihlášení žáka a učitele. Tyto přihlašovací údaje jsou přiděleny správcem při vytváření záznamu v databázi. Přihlašovací jméno tvoří první tři písmena ze jména, první tři z příjmení a pořadové číslo. Heslo je nastaveno jako rodné číslo a po přihlášení ho lze změnit na libovolnou kombinaci čísel.



Obrázek_11. Přihlášení do aplikace. (Zdroj: vlastní)

Po přihlášení se žákovi zobrazí jeho známky spolu s datem poslední úpravy a příjmením učitele, který změnu provedl (Obrázek_12). Vpravo nahoře je klasické tlačítko pro odhlášení a další pro změnu hesla. Také je zde zobrazeno jméno, příjmení a osobní číslo přihlášeného. Z bezpečnostních důvodů se aplikace automaticky po 10 minutách nečinnosti odhlásí. Pro pokračování v práci je nutné se znovu přihlásit.



Obrázek_12. Zobrazení známek žáka. (Zdroj: vlastní)

Sekce systému pro učitele je podstatně sofistikovanější. Základní vzhled a funkce jsou stejné jako u části systému pro žáka. Po přihlášení musí učitel vybrat ze seznamu třídu, ve které chce přidávat nebo editovat známky. Poté se zobrazí všichni žáci z dané třídy (Obrázek_13).

Obrázek_13. Zobrazení žáků ve vybrané třídě. (Zdroj: vlastní)

Dále je možno zobrazit žákovy známky, kárná opatření a pochvaly (Obrázek_14) nebo přímo přidat nové (Obrázek_15).

Právě pro jakoukoliv úpravu dat v databázi je potřeba vytvořit speciální metodu v Java Servletu a proceduru v SQL Serveru. Např. SQL kód na vytvoření procedury pro přidání známky do databáze vypadá takto:

```
CREATE PROC spPridatZnamku(
@OsobniCislo nchar(6),
@Znamka tinyint,
@KodPredmetu int,
@Poznamka nvarchar(100),
@Upravil nchar(6),
@Vytvoreno nchar(16))
AS
BEGIN
INSERT tblZnamky(OsobniCislo, Znamka, KodPredmetu, Poznamka,
Upravil, Vytvoreno)
```

```
VALUES (@OsobniCislo, @Znamka, @KodPredmetu, @Poznamka,
@Upravil, @Vytvoreno)
END
```

V Servletu je vytvořena metoda, ve které je volána výše uvedená procedura z SQL Serveru. Je nutné ji dodat také potřebné parametry z formuláře, který vyplnil učitel. Důležitá část této metody vypadá následovně:

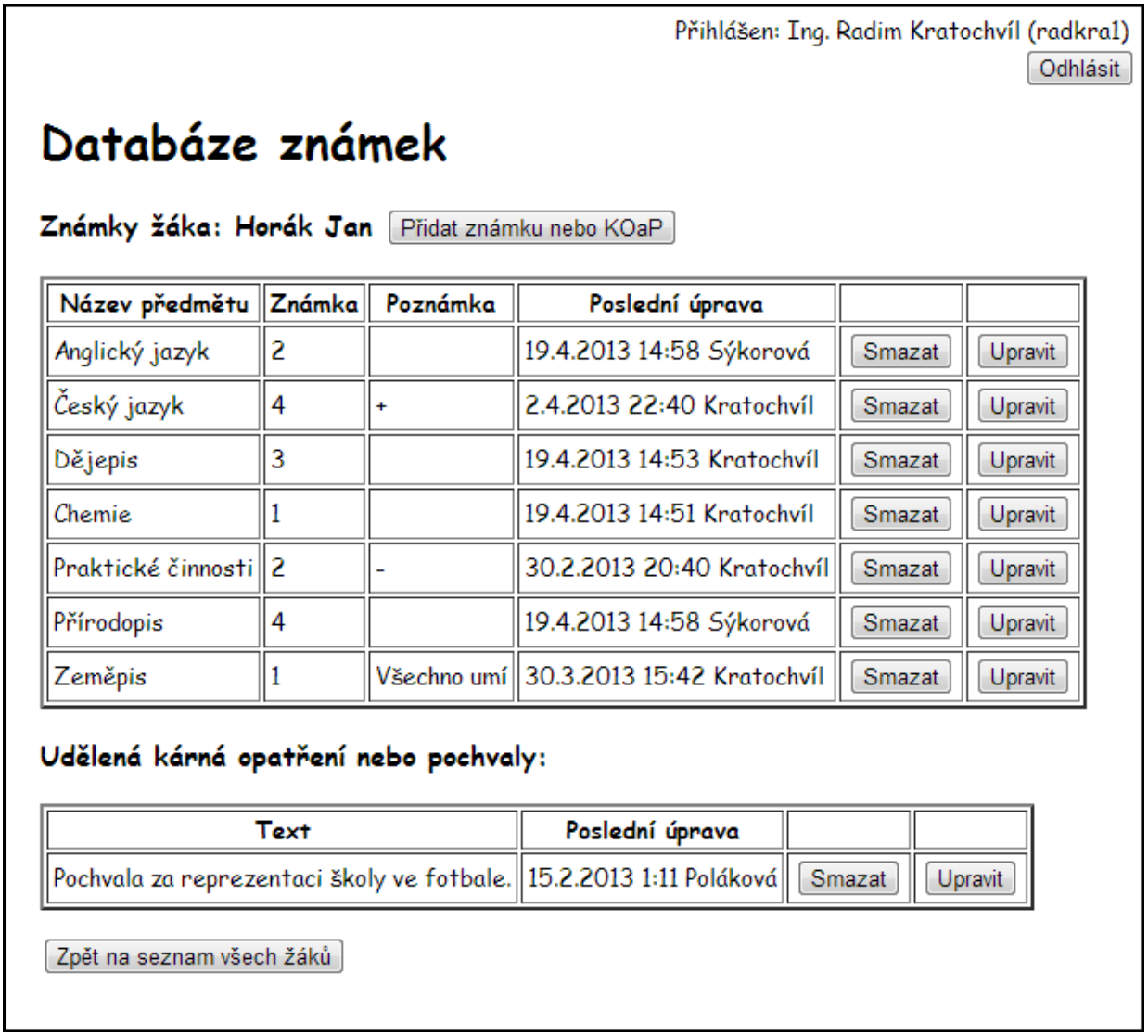
```
con = dbZnamky.getConnection();
        synchronized (con) {
           stat = con.prepareCall("{CALL spPridatZnamku(?,?,?,?,?,?)}");
           stat.setString(1, osobniCislo);
           stat.setInt(2, znamka);
           stat.setInt(3, kodpredmetu);
           stat.setString(4, poznamka);
           stat.setString(5, upravil);
           stat.setString(6, datum);
           stat.executeUpdate();
           request.setAttribute("novaznamka","Známka byla úspěšně přidána!");
           return "znamky.jsp";
        }
```

Uživatel by se mohl pokusit, přepsáním URL adresy nebo přepsáním některého používaného parametru, zobrazit stránky a údaje nebo dokonce provádět změny, na které nemá právo. Aby se tomuto neoprávněnému přístupu zabránilo, je před každým generování stránky automaticky kontrolováno právo daného přihlášení zobrazit tuto stránku. Tato kontrola spočívá v ověření, zda přihlášení (osobní číslo, pod kterým je daná osoba přihlášena) náleží do tabulky tblUcitele. V této tabulce jsou uloženy údaje o učitelích, kteří na rozdíl od žáků mají oprávnění k přidávání a změně dat. Použita je následující metoda v Java Servletu:

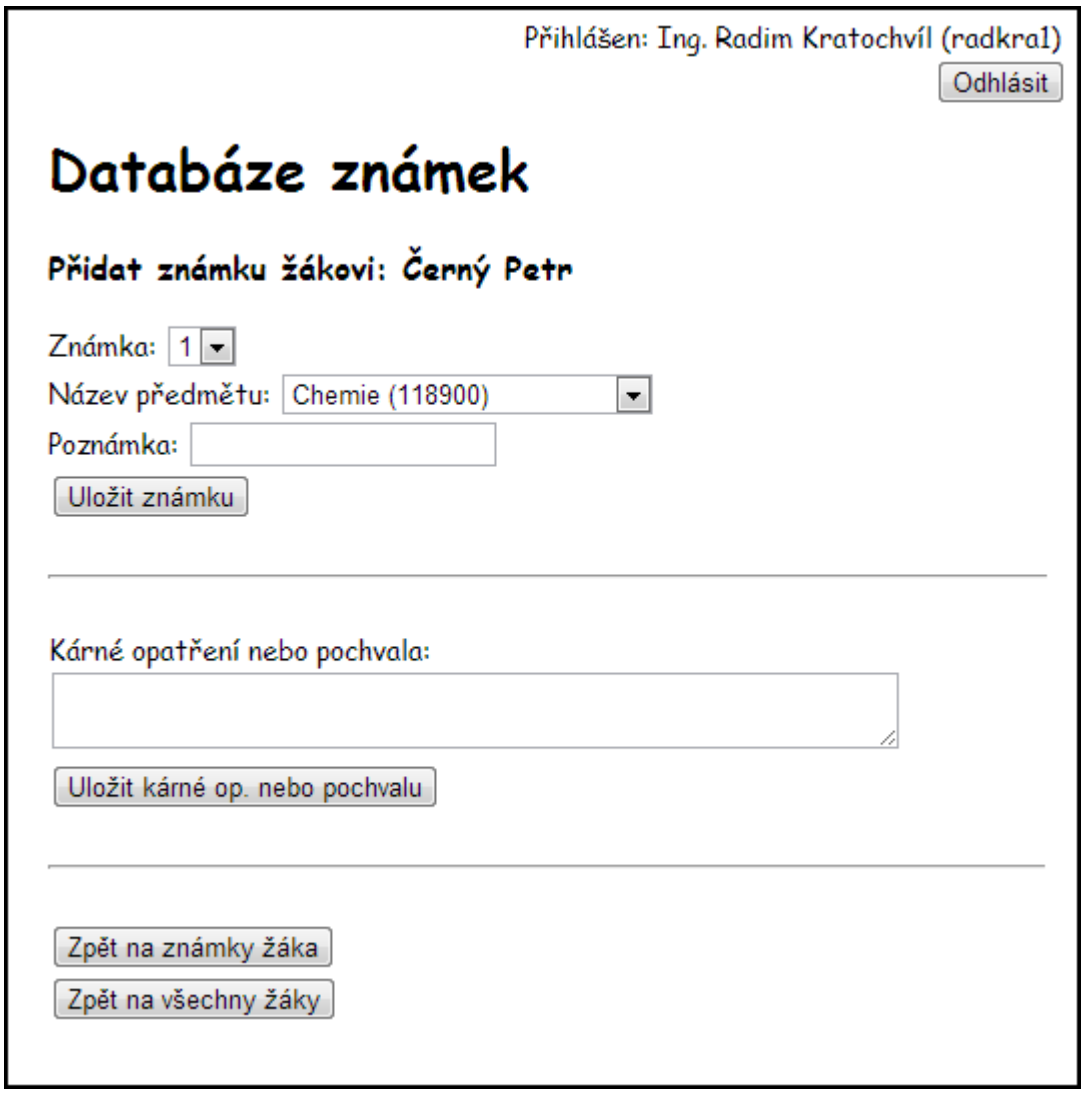
```
private String kontrola (HttpServletRequest request) {
        HttpSession session = request.getSession();
        String opr = (String)session.getAttribute("opravneni");
        if (opr.equals("ucitel")) {
                return "ok";
        } else {
```

```
                request.setAttribute("chybakontroly", "Nemáte oprávnění pro
                zobrazení této stránky!");
            return "pristupOdepren.jsp";
            }
    }
```

Metoda „kontrola“ získá ze session atribut s názvem oprávnění. Tento atribut je uložen automaticky při přihlášení a uživatel nemá možnost ovlivnit jeho hodnotu. Podle této hodnoty metoda vrací buď „ok“ když uživatel má příslušné oprávnění nebo pokud ho nemá, vrátí název stránky „pristupZamitnut.jsp“, na které se zobrazí chybová hláška a také má uživatel možnost se přihlásit pod jiným uživatelským jménem s příslušným oprávněním.



Obrázek_14. Zobrazení známek žáka učitelem. (Zdroj: vlastní)



Obrázek_15. Přidávání a úprava záznamů. (Zdroj: vlastní)

4 Zhodnocení navrhovaného řešení

V této kapitole jsou stručně představeny již existující nejznámější a nejrozšířenější komerční informační systémy pro školy Bakaláři, Škola OnLine, Etřídnice a Edookit. Také je zde provedeno porovnání se systémem vytvořeným v této práci a celkové zhodnocení.

Bakaláři

Tento systém je rozdělen na několik modulů a celkově patří mezi nejstarší. Jsou to: Evidence žáků a zaměstnanců, třídní kniha, webová aplikace (škola-rodiče), rozvrh hodin, suplování. Základní cena s webovou aplikací se pohybuje od 3300,- Kč až do cca 11000,- Kč podle zvolených doplňkových modulů. K ceně je nutno také přičíst částku za pravidelný každoroční update.

Evidence žáků a zaměstnanců *„zpracovává vedle osobních údajů zejména klasifikaci žáků. Z karty žáka pak lze vyčíst veškeré potřebné informace - osobní údaje, údaje o rodičích, kompletní klasifikaci za celou školní docházku a podobně. Velmi jednoduchý zápis známek zvládne i začátečník v práci s počítačem.“* (bakalari.cz, 2013)

Elektronická třídní kniha „umožňuje zápis jednotlivých hodin (číslo, téma hodiny, poznámky apod.), zadávání nepřítomnosti žáků v hodinách, omlouvání absence třídním učitelem, s možností tiskových výstupů v podobě původní třídní knihy.“ (bakalari.cz, 2013)

Webová aplikace zprostředkovává tyto funkce:

- *„Osobní údaje žáka - dle výběru, rodiče mohou nejen kontrolovat, zda škola eviduje správné údaje (telefony, adresy), systém umožňuje i přímé ohlašování změn přes www.*
- *Pololetní klasifikace - kompletní pololetní klasifikace ve všech ročnících od počátku studia, přehled zameškaných hodin atd.*
- *Průběžná klasifikace - přehled všech průběžně zadaných známek.*
- *Průběžná docházka - napojení na modul Třídní kniha - evidence zameškaných hodin - zobrazení po dnech, měsících, ale i přímo po vyučovacích hodinách.*
- *Výchovná opatření - přehled třídních důtek, pochval atd.“* (bakalari.cz, 2013)

Škola OnLine

„Škola OnLine je moderní školní informační systém, který umožňuje rychle a efektivně zpracovávat veškerou školní agendu při zachování vysokého uživatelského komfortu. Jedná se o webovou aplikaci, což znamená, že je dostupná 24 hodin denně prostřednictvím internetu, a to při využití pouze běžného webového prohlížeče bez nutnosti jakékoliv další instalace.“ (skolaonline.cz, 2013)

Tento systém zastřešuje velké množství rozšíření a doplňkových služeb. Mimo třídní knihu a hodnocení jsou to: Rozvrh a suplování, školní akce, tisk vysvědčení, výsledky přijímacích řízení, inventarizace majetku, správa zaměstnanců, propojení s docházkovým systémem, propojení se školní knihovnou a stravovacím systémem a další. Cena se pohybuje od 3500,- Kč až po 12 000,- Kč ročně, podle vybraných funkcionalit.

Etřídnice

„Etřídnice je jednoduchý, stabilní a komplexní školský informační systém. Aktuálně se skládá z modulů Elektronická třídní kniha, Elektronická žákovská knížka a Elektronický deník praxe - deník evidence odborného výcviku.

Elektronická třídní kniha - zcela nahradí papírovou třídní knihu. Umožňuje vedení záznamů o docházce, absencích, probrané látce, suplování a dalších informací. Tyto informace umožňuje sdílet s rodiči žáka a to v reálném čase. Slouží tedy nejen jako nástroj pro potřeby školy, ale také jako komunikační kanál školy směrem k rodičům.

Elektronická žákovská knížka - nahrazuje klasickou „žákovskou“. Umožňuje vést záznamy o prospěchu, absencích (spolupracuje s Elektronickou třídní knihou), poznámkách a dalších sděleních školy. Tyto informace umožňuje, podobně jako Elektronická třídní kniha, sdílet s rodiči žáka.“ (etridnice.cz, 2013)

Edookit

I tento informační systém je rozdělen na několik částí: Spolupráce, třídní kniha, osobní údaje, rozvrh, rodičovský portál a administrativa. Celý systém je provozován jako online aplikace, za kterou se zaplatí poplatek 2100,- Kč a měsíční částka cca 1000,- Kč.

Část, kterou vývojáři nazvali spolupráce, obsahuje tyto funkce: emailového klienta, kalendář, který lze sdílet s pracovní skupinou, souborový systém s možností sdílení dat, zadávání a kontrolu splnění úkolů a kontakty. Další částí je třídní kniha, ve které se zaznamenává docházka, hodnocení studentů, plánování tematického plánu hodin, zadávání

domácích úkolů, seznam žáků a učivo (tematické plány s možností vytváření online testů). Osobní údaje obsahují matriku žáků, seznam zaměstnanců, seznam rodičů a další záznamy. Rodičovský portál shrnuje informace o docházce žáků, hodnocení, učivu, domácích úkolech a událostech ve škole. Administrativní část uchovává evidenci knihovny, inventarizaci, informace o nákupech a objednávkách a evidenci přijatých plateb. (edookit.cz, 2013)

Porovnání systémů a zhodnocení

Jak je vidět, většina systémů je navržena jako webová aplikace, přes kterou jsou spravována všechna data. Ze zmíněných čtyř systémů pouze Bakaláři využívají webovou aplikaci ke klasifikaci atp. a zbytek systémů běží na školní síti a není dostupný z internetu. Všechny tyto systémy mají řadu volitelných doplňkových funkcí a služeb. Podle zvolených funkcí se poté odvíjí cena za komplet.

Určitou nevýhodou těchto systémů může být, např. při zapisování absence v jednotlivých hodinách do systému, nutnost přítomnosti počítače nebo notebooku v každé učebně. To je hlavně pro menší školy finančně neúnosné. Samozřejmě je zde možnost zápisu záznamů na papír a poté jejich uložení do systému. Tím ovšem vzniká větší administrativní zátěž učitelů.

Aplikace vytvořená v této práci zohledňuje současný trend a tudíž je koncipována tak, aby byla dostupná z jakéhokoliv počítače připojeného do sítě internet. To byl také v podstatě základní požadavek firmy Xevos Solutions s.r.o., se kterou byla tato práce konzultována a s níž probíhala spolupráce při vývoji systému.

Komponenty informačního systému (databáze i zdrojové kódy webové aplikace) ve finální verzi byly vypáleny na CD, které je přiloženo na zadní straně této práce (příloha_1). Vše je připraveno pro nahrání na server a v nejbližší době bude spuštěn testovací provoz.

5 Závěr

Cílem této práce bylo vytvořit informační systém, pro menší základní školu. Tento systém měl podporovat informovanost a usnadňovat sdílení informací o studijních výsledcích žáků mezi školou a rodinami žáků. Z tohoto důvodu měl být koncipován tak aby byl přístupný odkudkoliv a kdykoliv, tudíž jako webová aplikace, která je propojena s databází.

V teoretické části se práce snaží přiblížit a vysvětlit problematiku informačních systémů jako celku i jeho dílčích částí. Nejprve byla zmíněna historie těchto IS a byly objasněny pojmy databáze, strukturovaný dotazovací jazyk, technologie Java Servlet, JavaServer Pages atp. Ke všem použitým pojmům byla uvedena teorie nezbytná pro jejich upřesnění a správné pochopení.

Další kapitola pojednává o praktickém řešení. Nejdříve byl v konceptuálním návrhu definován systém a vytvořen případ užití pro zjištění všech požadavků na databázi a webovou aplikaci. Dále byly definovány entity databáze a jejich kardinalita vztahů, vytvořen důležitý entitně-relační diagram a konceptuální návrh aplikace. Logický návrh byl zaměřen na využití ER diagramu pro vytvoření návrhu tabulek a ne podrobný popis všech sloupců a vztahů mezi tabulkami v databázi. V poslední části této kapitoly byla s využitím všech předchozích poznatků vytvořena funkční databáze v Microsoft SQL Serveru a webová aplikace v prostředí NetBeans IDE.

V poslední kapitole byly představeny již existující informační systémy pro základní a střední školy. Bylo zde také provedeno srovnání se systémem navrhnutým a vytvořeným v této práci.

Všechny cíle, včetně průběžných, které vyvstaly během zpracování praktické části nebo při konzultacích ve firmě, byly splněny. Navrhnuté řešení splňuje všechny požadavky zadavatele. Systém byl otestován a je připraven pro uvedení do ostrého provozu. Je ovšem potřeba vzít v úvahu fakt, že právě při běhu systému v praxi se mohou objevit nedostatky, které si mohou žádat dílčí úpravy. Další vývoj systému bude probíhat podle specifických požadavků konkrétní školy. Například přidání některé doplňkové funkce nebo vytvoření aplikace v Microsoft Acces pro pohodlnější a jednodušší správu dat v databázi.

Seznam použité literatury

Odborná literatura

BRUCKNER, Tomáš a kolektiv. *Tvorba informačních systémů: principy, metodiky, architektury*. Praha: Grada, 2012. ISBN 978-80-247-4153-6.

CONOLLY, T., C. BEGG a R. HOLOWCZAK. *Mistrovství - databáze: profesionální průvodce tvorbou efektivních databází.* Brno: Computer Press, 2009.
ISBN 978-80-251-2328-7.

KISZKA, Bogdan. *1001 tipů a triků pro jazyk Java*. Brno: Computer press, 2009. ISBN 978-80-251-2467-3.

OPPEL, Andy. *SQL bez předchozích znalostí.* Brno: Computer press, 2008. ISBN 978-80-251-1707-1.

TVRDÍKOVÁ, Milena. *Aplikace moderních informačních technologií v řízení firmy*. Praha: Grada, 2008. ISBN 978-80-247-2728-8.

VOŘÍŠEK, Jiří. *Strategické řízení informačního systému a systémová integrace.* Praha: Management Press, 1997. ISBN 80-85943-40-9.

Elektronické dokumenty a ostatní

bakalari.cz *Evidence, klasifikace, vysvědčení*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.bakalari.cz/evidence.aspx.

bakalari.cz *Třídní kniha*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.bakalari.cz/trkniha.aspx.

bakalari.cz *Webové aplikace*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.bakalari.cz/webapp.aspx.

bakalari.cz *Ceník programů*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.bakalari.cz/cenyprog.htm.

etridnice.cz *Informační systém pro školy*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.etridnice.cz.

edookit.cz *Informační systém pro základní a střední školy*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.edookit.cz/funkce.html.

edookit.cz *Ceník.* [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.edookit.cz/cenik.html.

java.net *GlassFish Server Open Source edition 3.1 Quick Start Guide* [online]. 2013, [cit. 2013-05-02]. Dostupné z: http://www.glassfish.java.net/docs/3.1/quick-start-guide.pdf

microsoft.com *Edice Microsoft SQL Server Express*. [online]. 2013, [cit. 2013-05-02]. Dostupné z: http://www.microsoft.com/sqlserver/cs/cz/editions/express.aspx

netbeans.org *Vývojové prostředí NetBeans IDE*. [online]. 2013, [cit. 2013-05-02]. Dostupné z: http://www.netbeans.org/index_cs.html

oracle.com *What is an Application Server*. [online]. 2013, [cit. 2013-05-02]. Dostupné z: http://www.wikis.oracle.com/display/GlassFish/FaqWhatIsAppServer.

skolaonline.cz *Školní informační systém Škola OnLine*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.skolaonline.cz/Skolni_informacni_system.aspx.

skolaonline.cz *Ceník školního informačního systému*. [online]. 2013, [cit. 2013-04-19]. Dostupné z: http://www.skolaonline.cz/Reditel/Cenik.aspx.

SKŘIVAN, Jaromír. *Databáze a jazyk SQL*. [online]. 2000, [cit. 2013-04-25]. Dostupné z: http://interval.cz/clanky/databaze-a-jazyk-sql.

tutorialspoint.com *JSP - Standard Tag Library (JSTL) Tutorial*. [online]. 2013, [cit. 2013-05-07]. Dostupné z: http://www.tutorialspoint.com/jsp/jsp_standard_tag_library.htm.

Seznam zkratek

Apod. – a podobně
DA – Data Administrator (správce dat)
DBA – Database Administrator (správce databáze)
DBMS - Database Management System (systém řízení báze dat)
BCNF - Boyce-Coddova normální forma
DCL – Data Control Language (jazyk pro kontrolu dat)
DDL - Data Definition Language (jazyk pro definici datových struktur)
DML - Data Manipulation Language (jazyk pro manipulaci s daty)
EL - Expression Language
ER model – entitně-relační model
FK – Foreign Key (cizí klíč)
Html - Hypertext Markup Language
Http – Hypertext Transfer Protocol
ICT - Information and Communication Technology (informační a komunikační technologie)
IDE - Integrated Development Environment (vývojové prostředí)
IEC - International Electrotechnical Commision (Mezinárodní úřad pro elektrotechniku)
IS - Information System (informační systém)
ISO - International Organization for Standardization (Mezinárodní organizace pro standardizaci)
JSP - JavaServer Pages
JSTL - JSP Standard Tag Library
Např. – například
NF – normální forma
OODBMS - Object-Oriented Database Management System (objektově orientované systémy řízení databáze)
Op. – opatření
ORDBMS - Object-Relational Database Management System (objektově relační systémy řízení databáze)
PC – Personal Computer (osobní počítač)
PK – Primary Key (primární klíč)
RDBMS - Relational Database Management System (relační systém řízení databáze)

SQL - Structured Query Language (Strukturovaný dotazovací jazyk)

Tzn. – to znamená

Tzv. – takzvaně

URI - Universal Resource Identifier

URL - Uniform Resource Locators

Prohlašuji, že

- Jsem byl seznámen s tím, že na mou diplomovou (bakalářskou) práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo;
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně, ke své vnitřní potřebě, diplomovou (bakalářskou) práci užít (§ 35 odst. 3);
- souhlasím s tím, že diplomová (bakalářská) práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO a jeden výtisk bude uložen u vedoucího diplomové (bakalářské) práce. Souhlasím s tím, že bibliografické údaje o diplomové (bakalářské) práci budou zveřejněny v informačním systému VŠB-TUO;
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;
- bylo sjednáno, že užít své dílo, diplomovou (bakalářskou) práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Ostravě dne 10.5.2013

.....

Václav Koběrský